Peer-to-Peer Rating*

Danny Bickson
The Hebrew University of Jerusalem

Dahlia Malkhi, Lidong Zhou
Microsoft Research, Silicon Valley

Abstract

This paper proposes to utilize algorithms from the probabilistic graphical models domain for Peer-to-Peer rating of data items and for computing "social influence" of nodes in a Peer-to-peer social network.

We evaluate the practicality of our approach using large-scale simulations over a MSN Live Messenger subgraph consisting of about a million nodes.

Our algorithms are general since they can be used for Peer-to-peer monitoring and for the efficient computation of other node ranking methods, such as PageRank and Information Centrality.

1 Introduction

Whether you are browsing for a hotel, searching the web, or looking for a recommendation on a local doctor, what your friends like will bear great significance for you. This vision of virtual social communities underlies the stellar success of a growing body of recent web services, e.g., `http://www.flickr.com`, `http://del.icio.us`, `http://www.myspace.com`, and others. However, while all of these services are centralized, the full flexibility of social information-sharing can often be better realized through direct sharing between peers.

This paper presents a mechanism for sharing user *ratings* (e.g., on movies, doctors, and vendors) in a social network. It introduces distributed mechanisms for computing by the network itself individual ratings that incorporate rating-information from the network. Our approach utilizes message-passing algorithms from the domain of Gaussian graphical models. In our system, information remains in the network, and is never "shipped" to a centralized server for any global computation. Our algorithms provide each user in the network with an individualized rating per object (e.g., per vendor). The end-result is a local rating per user which minimizes her cost function from her own rating (if exists) and, at the same time, benefits from incorporating ratings from her network vicinity. Our rating converges quickly to an approximate optimal value even in sizable networks.

More specifically, consider the following settings. Suppose that regarding a specific issue, say, a movie, every user has an initial opinion represented as a value in some range (or a special empty value $\perp$). Further assume that users are related to others through social connections. Then the goal is to utilize each user's own rating as well as to integrate ratings from the social network for providing the user with an individualized recommendation. The first challenge is to define a utility target that delivers a sensible personalized result to each user. We achieve this by forming a bi-criterion cost function that combines the user's own rating (if exists) and the ratings of its immediate social contacts. The goal is to minimize the square differences in both components over the entire system.

The next challenge is to devise a distributed algorithm that calculates ratings at each user. We employ a recent algorithm for *consensus propagation* [1] that provably converges to values that minimize our cost function. The algorithm is purely decentralized and works iteratively: In each round, a node integrates the values it received from its immediate neighbors in the previous round, and sends a new value to the neighbors. Previously, it was not known how this algorithm would perform in practice in terms of convergence time. Using simulations, we demonstrate fast convergence, as seen in the body of the paper. To the best of our knowledge, our simulation is the first in the field of belief propagation to attempt a network of millions of nodes.

The final challenge we face is to assess the effectiveness of our rating system. This requires developing some sort of benchmark that tells whether the rating is "good", and is potentially of independent interest. We devised the following experiment. We first produce an influence-ranking of the nodes. Rather than using PageRank or HITS ranking, we again let the network itself compute a ranking in a distributed manner. We then examine the effects that influential (high-ranking) nodes have on ratings of users in the network. The result we expect from a good rating system is that the ratings of influential nodes will propagate and bias the ratings of significant portions of the network,

*Danny Bickson was partially supported by Microsoft Research Internship and by EVERGROW IP 1935 of the EU Sixth Framework.

and would quench any bogus ratings inserted by poorly-connected nodes.

We ran our benchmark over real (anonymized) data taken from the Windows Live Messenger network. Each node represents a Messenger user, and edges connect nodes with those on their buddy lists. Our findings are as follows. The ratings of high-ranking nodes have clearly dominated the influence of low-ranking nodes. Moreover, even when vastly outnumbered, high ranking nodes still biased the rating significantly more than low ranking nodes. These results are encouraging indications that the rating system indeed provides sensible rating values.

Due to its complete decentralization, and its good scalability, our work is suitable for deployment in social networks such as Messenger and others. It has been developed as part of the Nocturnal project [21]. Nocturnal provides an infrastructure for sharing views/recommendations among social communities using an existing instant-messaging network, without a central server. The core idea is "gossip": I share my views directly with my Messenger contacts, they pass my views to their contacts, and so on. Every user in the network stores this gossip data locally on his/her disk.

Sharing views over a social network has many advantages. By taking a peer-to-peer approach the user information is shared and stored only within its community. Thus, there is no need for trusted centralized authority, which could be biased by economic and/or political forces. Likewise, a user can constrain information pulling from its trusted social circle. This prevents spammers from penetrating the system with false recommendations.

The structure of this paper is as follows: in Section 1.1 we briefly outline the related work regarding both ranking of data items and ranking of nodes in a social network. In Section 2 we define the data rating problem and propose our solution in Section 3. We present simulation results of a subgraph cut from a real social network topology, in Section 4. We discuss several protocol extensions in Section 5. We conclude in Section 6.

1.1 Related work

There is a mass of related work in the learning field regarding rating of data items. For example [12], [11] ranks movie recommendation to users based on their past rating of movies. Both papers achieve very good prediction results based on user movie rating databases. The problem is that the algorithms assume centralized execution and global knowledge of the network.

In Peer-to-peer settings, several works address collaborative filtering [14]. Recent work [13] recommends files to users in a file sharing network. Awerbuch et al. discusses in [17] how to find items for matching user tastes. Those works assume a binary settings when one of more items should be recommended to users.

We propose to utilize algorithms from the probabilistic graphical models domain in Peer-to-peer settings. We chose to address the simple question of rating a single data item in the network, where the rating is real number (not a binary recommendation) for showing the applicability of our technique. Rating of multiple data items can be done by running our algorithm in parallel.

Other works discuss the influence of nodes in a social network [16, 15]. A good overview of the existing approaches is given in [16]. There are several known ways to compute node importance in networks, for example Google's PageRank [5] algorithm, HITS [9] and EigenTrust [8]. Another interesting method is the electrical conductivity (EC) method [19] which regards the social network as an electrical network and measures the social influence by calculating the electrical current flow.

Our work related to personalized rating of data items can be easily utilized in other domains as well. For example, for monitoring. Each node has a certain property we would like to monitor, like the node current local load. Computing a global load in a very large and dynamic Peer-to-peer network is not always meaningful. However, computing the load in a certain vicinity of a node, can be easily done using our technique.

We propose a novel method called "Spatial Ranking" to compute a personalized ranking of the nodes in the social network. Our method is based on the observation that when a node i calculates a personalized ranking of the network, it gives greater significance to nodes that are found in its vicinity. Since social influence is a heuristic calculation, we do not claim that our technique is better than other methods. Different ranking methods capture different aspects of the social network. The method we show is general since it can be used other ranking methods as well. Specifically, we show how to compute PageRank and Information Centrality ranking methods, using the our algorithm.

2 The Data Rating Problem

The social network is represented by an directed, weighted communication graph $G = (V, E)$. Nodes $V = \{1, 2, ..., n\}$ are users. Edges express social ties, where an edge weight w_{ij} indicates a measure of the mutual trust between the endpoint nodes i and j. A node i's neighbors are denoted by $N(i)$.

We consider a single instance of the rating problem that concerns an individual item (e.g., a movie). In practice, the system maintains ratings of many objects concurrently, but presently, we do not discuss any correlations across items. Concerning one item, each node i has an initial input rating y_i, which may potentially be the special empty value $\perp$. The output of the rating algorithm provides every node i

with a final rating x_i. The "goodness" of the rating vector is expressed by the cost function below.

2.1 The Proposed Cost Function

Our proposed cost function is composed of two terms. The first one — $(x_i - y_i)^2$ — measures the change in user rating from their initial values. The second term — $\Sigma_{j \in N(i)} w_{ij}(x_i - x_j)^2$ — measures the difference between a user's rating and its buddies' rating. Naturally, these terms may be contradicting. A parameter β determines the relative importance among the two terms. When β is very large, the nodes will have a good coordination with their neighbors. When β is small, node will take into account their own value with higher importance.

The goal is to find a vector that minimizes the overall cost of both terms throughout the network, i.e., calculate the following minimum:

$$min_x[\Sigma_{i \in V}(x_i - y_i)^2 + \beta \Sigma_{w_{ij} \in E} w_{ij}(x_i - x_j)^2] \quad (1)$$

Note, that the above cost function is general in the sense it can be utilized for other useful computations in the Peer-to-Peer network. One immediate example is for monitoring some nodes' parameters like the network load. In this case, each y_i is the current load the node is experiencing, x_i is the output estimated load in the node's vicinity. The parameter β defines the level of correlation between node values, where a very large β will force the output values to converge to the average load. The weights w_{ij} define the relations between nodes, where closer nodes will have higher weights, forcing the load to converge locally in each vicinity.

3 Peer-to-Peer Rating

We first give a brief background of Gauss-Markov Random field (GMRF).

GMRF Background. A Gauss-Markov Random field is a Markov Random Field where the joint distribution is Gaussian $p(x) \sim N(\mu, P)$. There are two parameters which characterize the distribution: the mean μ is a vector of length n. The covariance matrix P is matrix of size $n \times n$ defined by $P_{ij} \equiv Cov(X_i, X_j) = E\{(x - \mu_i)(x - \mu_j)\}$. The probability density function is given in the following form:

$$p(x) = \alpha \exp\{-\frac{1}{2}(x - \mu)' P^{-1}(x - \mu)\}$$

where α is a normalization factor. An alternative parametrization is provided by the information form specified in terms of a *field vector* $\mu = P^{-1}h$ and an *information matrix* $J = P^{-1}$. The pdf of the process is:

$$p(x) = \beta \exp\{\frac{1}{2}x' Jx + h'x\}.$$

The *inference problem* is defined as follows [7]: Given a graphical model of the GMRF in the information filter form (h, J), evaluate the marginal densities $x_i = N(\mu_i, P_{ii})$ for all nodes i. Note that this corresponds to recovering the mean vector μ and the diagonal of the covariance matrix P.

We model our social network as a GMRF, where each user is a node in the graph, and each edge is a social link between users. Each node represents a scalar Gaussian distribution of mean y_i (its prior input rating). The matrix W of edge weights (trust values) is used as the covariance information matrix. We set the matrix diagonal values to 1, which can be interpreted as giving equal "weights" to the initial ratings of all users.

The output of the calculation is the solution for the inference problem: we would like to calculate for each node the marginal probability $p(x_i) = N(\mu_i, P_{ii}^{-1})$. The mean μ_i of the Gaussian x_i is the output rating.

We utilize the Consensus Propagation (CP) algorithm [1] to solve the above minimization problem. The algorithm is a variant of the belief propagation algorithm over a Gauss-Markov random field (GMRF). It is a distributed message passing algorithm where the interaction is between neighboring nodes. Each node locally computes its output, the rating. The algorithm is proved to converge towards the optimal value which minimizes the proposed cost function given in (1).

The Consensus Propagation Algorithm (CP) [1] is a distributed algorithm for calculating the network average, assuming each node has an initial value. We have extended the CP algorithm in several ways. First, in the original paper the authors propose to use a very large β for calculating the network average. As mentioned, large β forces all the nodes to converge to the same value. We remove this restriction by allowing a flexible selection of β based on the application needs. As β becomes small the calculation is done in a closer and closer vicinity of the node. In the extreme case where β is zero, the calculation is done only locally in the node, without taking into account its neighbors values.

Second, we have extended the CP algorithm to support null value, adapting it to omit the term $(y_i - x_i)^2$ when $y_i = \perp$, i.e., when node i has no initial rating.

Third, we use non-uniform edge weights w_{ij}, which in our settings represent mutual trust among nodes. This makes the rating local to certain vicinities, since we believe there is no meaning for getting a global rating in a very large network. This extension allows also asymmetric links where the trust assigned between neighbors is not symmetric.

The algorithm computes the following in a loop:

$$J_{i\to j} = (\frac{1}{(1+\Sigma_{k\in N(i)\setminus j} J_{k\to i})} + \frac{1}{\beta w_{ij}})^{-1}$$

$$\mu_{i\to j} = \frac{y_i + \Sigma_{k\in N(i)\setminus j}\ J_{k\to i}\mu_{k\to i}}{1+\Sigma_{k\in N(i)\setminus j} J_{k\to i}}$$

$$x_i = \frac{y_i + \Sigma_{k\in N(i)} J_{k\to i}\mu_{k\to i}}{1+\Sigma_{k\in N(i)} J_{k\to i}}$$

Where a message from node i to node j has two real numbers: The precision (one over the variance) $J_{i\to j}$ and the mean $\mu_{i\to j}$. $N(i)$ is the list of node i's neighbors, β is the weighting constant discussed earlier and y_i is the prior input. x_i is the posterior, the output computed locally in each node.

Convergence can be detected locally by measuring the change between the previous rating value $x_i^{(t-1)}$ and the current value $x_i^{(t)}$. Rounds do not need to be synchronized. Recent work shows that in most cases when the rounds are no synchronized, the algorithm converges faster [10]. The algorithm terminates locally when a node detects that there is no change in the local decision value x_i.

Efficiency. The local computation at a node at each round is fairly minimal. Each node i computes locally two scalar values $\mu_{i\to j}$, $J_{i\to j}$ for each neighbor $j \in N(i)$. Convergence time is dependent on the network topology. Empirical results are provided in the next section.

3.1 Rating Convergence

For evaluating the feasibility and efficiency of our rating system, we used anonymized data obtained from Windows Live Messenger that represents Messenger users' buddy relationships. The Messenger network is rather large for simulation (over two hundred million users), and so we cut subgraphs by starting at a random node, and taking a BFS cut of about one million nodes. We repeated this experiment to get several such networks with different scales, up to several millions nodes. The diameter of the sampled graph is five on average. Note that, we do not use the full Messenger graph in a centralized simulation because of its enormous size. In a distributed network settings, the rating algorithm can work even on such a large network. Furthermore, we expect that convergence time will have a similar magnitude on the full graph. (Convergence speed is related to the graph diameter which is very low on average in the full graph).

The experiment is geared towards evaluating the convergence speed of our rating method in the Messenger social network's settings. Therefore, in this test, input ratings y_i and edge weights w_{ij} are simply drawn uniformly at random in the range $[0-1]$. We have repeated this experiment with different initializations for the input rating and the edge weights and got similar convergence results.

Figure 1 shows the convergence speed. The x-axis represents round numbers[1]. The y-axis represents the sum-total of change in ratings relative to the previous round. We can see that the node ratings converge very fast towards the optimal rating derived from the cost function. After only five iterations, the total change in nodes ratings is about 1 (which means an average change of 1×10^{-6} per node).

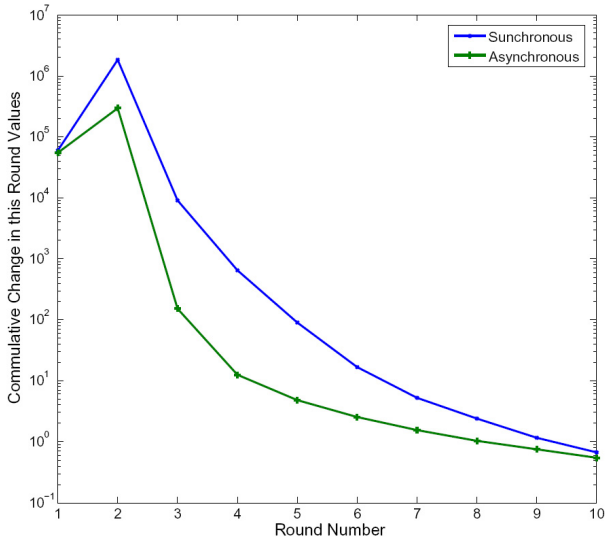

Figure 1. *Convergence of rating over a social network of 800,000 nodes and 9,000,000 edges. Note, that using asynchronous rounds, the algorithm converges faster, as discussed in [10]*

4 Rating Benchmark

One of the interesting challenges we faced was the need to come up with a "benchmark" that evaluates the effectiveness of our rating system. Demonstrating this requires a quantitative measure beyond mere speed and scalability. The benchmark approach we take is as follows. First, we produce a ladder of "social influence" that is inferred from the network topology, and rank nodes by this ladder. Next, we test our rating method in the following settings. Some nodes are initialized with rate values, while other nodes are initialized with empty ratings. Influential nodes are given different initial ratings than non-influential nodes. The expected result is that the ratings of influential nodes should

[1]The rounds are given only for reference, in practice there is no need for the nodes to be synchronized in rounds.

affect the ratings of the rest of the network so long as they are not vastly outnumbered by opposite ratings.

As a remark, we note that we can use the social ranks additionally as trust indicators, giving higher trust values to edges which are incident to high ranking nodes, and vice versa. This has the nice effect of initially placing low trust on intruders, which by assumption, cannot appear influential.

4.1 Spatial Ranking

In this section we explain how we rank the socially influential nodes in the social network for testing our rating method.

Our contributions regarding node ranking in the network are twofold: First, we present a novel algorithm to compute personalized ranking of nodes in the network. Second, we show that our algorithm is general since it is able to compute distributively and efficiently many of the existing ranking methods including PageRank [5] and Information centrality [19].

The "Spatial Ranking" algorithm output is not a global ranking which outputs one rank list of all the nodes in the network, but a personalized ranking which may be different for each node. In our method, each node ranks itself the list of all other nodes based on its network topology and creates a personalized ranking of its own.

We do not claim that our ranking is better or worse than other methods since "socially influential" is a heuristic computation. There are many different methods for computing social influence and each of them is capturing some different aspects of the social network properties. To this end, we show that our method can calculate efficiently other ranking methods.

4.1.1 Spatial Ranking Details

We propose to model the network as a Markov chain with a transition matrix R, and calculate the *fundamental matrix* P, where the entry P_{ij} is the expected number of times of a random walk starting from node i visits node j [4].

We take the local value P_{ii} of the fundamental matrix P, computed in node i, as node i's global importance. Intuitively, the value signifies the weight of infinite number of random walks that start and end in node i. Unlike the PageRank ranking method, we explicitly bias the computation towards node i, since we force the random walks to start from it. This captures the local importance node i has when we compute a personalized rating of the network locally by node i. Figure 2 captures this bias using a simple network of 10 nodes.

In Section 5 we show how to compute the full personal ranking of nodes (the full matrix P), where each node has a personal ranking of all the other nodes.

The fundamental matrix can be calculated by summing the expectations of random walks of length one, two, three etc., $R + R^2 + R^3 + \ldots$. Assuming that the spectral radius $\varrho(R) < 1$, we get $\sum_{l=1}^{\infty} R^l = (I - R)^{-1}$. Since R is stochastic, the inverse $(I - R)^{-1}$ does not exist. We therefore slightly change the problem: we select a parameter $\alpha < 1$, to initialize the matrix $J = I - \alpha R$ where I is the identity matrix. We know that $\varrho(\alpha R) < 1$ and thus $\alpha R + \alpha^2 R^2 + \alpha^3 R^3 + \ldots$ converges to $(I - \alpha R)^{-1}$. We use the message passing formulation of the Gaussian Belief Propagation (GaBP) algorithm given in [2] for computing $(I - \alpha R)^{-1}$ of the social network, by the network itself:

Input: Adjacency matrix J (the correlation matrix), shift vector h
Iterate:

$$h_{i\backslash j} = h_i + \sum_{k \in N(i)\backslash j} \Delta h_{k\to i} \; , \; J_{i\backslash j} = J_{ii} + \sum_{k \in N(i)\backslash j} \Delta J_{k\to i} \tag{2}$$

$$\Delta h_{i\to j} = -J_{ji}(J_{i\backslash j})^{-1} h_{i\backslash j} \; , \; \Delta J_{i\to j} = -J_{ji}(J_{i\backslash j})^{-1} J_{ij} \tag{3}$$

Where $h_{i\backslash j}, J_{i\backslash j}$ are intermediate terms calculated by the nodes, and each message sent from node i to node j is composed of two reals: $\Delta J_{i\to j}, \Delta h_{i\to j}$, the first represents the Gaussian precision (one over the variance) and the second represents the Gaussian mean. Finally, each node computes locally:

$$\mu_i = h_i + \sum_{k \in N(i)} \Delta h_{k\to i} \; , \; J_i = J_{ii} + \sum_{k \in N(i)} \Delta J_{k\to i}$$

Convergence can be detected locally when there is no change between messages in consecutive rounds.

The convergence of the algorithm is guaranteed, since we carefully selected $\alpha < 1$ s.t. $\varrho(\alpha R) < 1$, it is known that the BP algorithm over GMRF converges to the right mean, and the variances calculated are an approximation. We use the variances of each node as the spatial ranking of that node. An analysis of the variance error can be found in [6].

The BP convergence proofs assumes that the BP algorithm operates in synchronous rounds. The assumption of synchronized clocks is not reasonable in a very large social network.

4.2 Experimental results

For performing our benchmark tests, we once again used simulation over the ~ 1 million node sub-graph of the Messenger network. Using the ranks produced by our spatial ranking, we selected the seven highest ranking nodes and assigned them an initial rating value 5. We also selected seven of the lowest ranking nodes and initialized them with

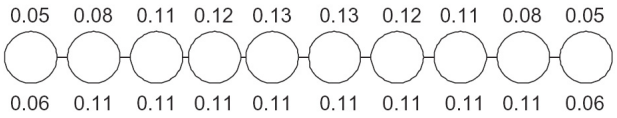

Figure 2. *Example output of the Spatial ranking (on top) vs. PageRank (bottom) over a network of 10 nodes. In the Spatial ranking method node rank is biased towards the center, where in the PageRank method, non-leaf nodes have equal rank. This can be explained by the fact that the sum of self-returning random walks increases towards the center.*

rating value 1. All other nodes started with null input. The results of the rating system in this settings are given in Figure 3. After about ten rounds, a majority of the nodes converged to a rating very close to the one proposed by the influential nodes. We ran a variety of similar tests and obtained similar results in all cases where the influential nodes were not totally outnumbered by opposite initial ratings; for brevity, we report only one such test here.

The conclusion we draw from these test is that our rating system provides sensible rating values to users. Quite importantly, it provides good protection against malicious infiltrators: Assuming that intruders have low connectivity to the rest of the network, we demonstrate that it is hard for them to influence the rating values in the system. Furthermore, we note that this property will be reinforced if the trust values on edges are reduced due to their low ranks, and using users satisfaction feedback.

5 Protocol Extensions

5.1 Extending the technique for computing the full fundamental matrix

We propose to extend the Walk-Sum technique [2] to compute the full matrix P (including the off-diagonal entries) to have a full ranking of nodes in the network. In the extended method the output of the local calculation of node i is the row $P_{[i]}$, where column j signifies the importance of node j to node i. We use the conditional correlation Lemma given in [6] for computing in parallel the full matrix P. The trick is to run n instances of GaBP in parallel, where in instance i the shift vector h_i in node i is initialized to one, and all other shift vectors are initialized to zero. Note that using this method, the computation of matrix P is exact and not an approximation as was done in [2].

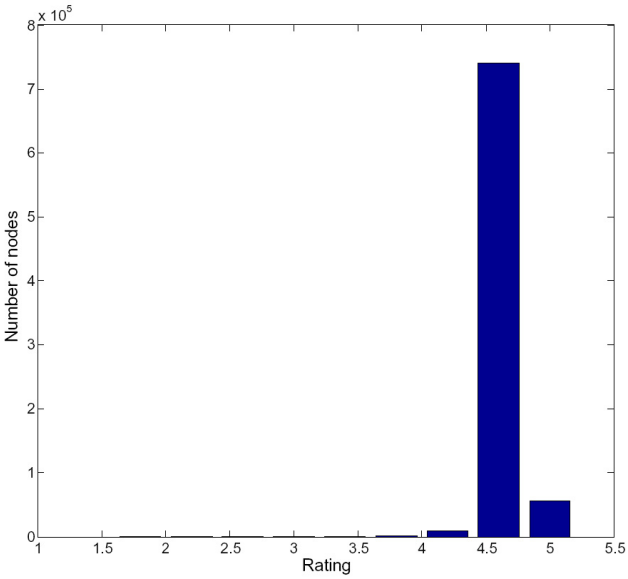

Figure 3. *Final rating values in a network of 800,000 nodes. Initially,* 7 *highest ranking nodes rate* 5 *and* 7 *lowest ranking nodes rate* 1.

5.2 Extending our technique to compute other ranking methods

PageRank. The PageRank algorithm is a fundamental algorithms in computing node ranks in the network [5]. In a nutshell, the Markov chain transition matrix M is constructed out of the web links graph. A prior probability x can be taken to weight the result. The personalized PageRank calculation can be computed [20]:

$$PR(x) = (1-\alpha)(I - \alpha M^T)^{-1}x$$

Where α is a weighting constant which determines the speed of convergence in trade-off with the accuracy of the solution and I is the identity matrix.

We propose to compute the personalized PageRank using our GaBP algorithm. The input to the algorithm is the correlation matrix $J = (I - \alpha M^T)$ and the shift vector $h = x$. The output is the vector $(I - \alpha M^T)^{-1}x$. Each node stores the result of its own personalized PageRank.

Information Centrality. In the information centrality node ranking method [19], the non-negative weighted graph $G = (V, E)$ is considered as an electrical network, where edge weights is taken to be the electrical conductance. A vector of supply b is given as input, and the question is to compute the electrical potentials vector p. This is done by computing the graph Laplacian and solving the set of linear equations $Lp = b$. The information

centrality method (a.k.a current flow betweenes centrality) is defined by:

$$IC(i) = \frac{n-1}{\Sigma_{i \neq j} p_{ij}(i) - p_{ij}(j)}$$

The motivation behind this definition is that a centrality of node is measured by inverse proportion to the effective resistance between a node to all other nodes. In case the effective resistance is lower, there is a higher electrical current flow in the network, thus making the node more "socially influental".

The relation between our algorithm and this proposed method, is that [19] shows that the information centrality can be calculated using $(L + J)^{-1}$ where L is the graph Laplacian and J is a matrix of all ones. So we can initialize our algorithm with $(L + J)$ matrix as input, and compute its inverse, same as we did in the previous case. The output in this case is that each node holds its own information centrality computed locally.

6 Conclusion

The rating system presented here achieves the following design goals.

1. The rating algorithm is completely decentralized and uses the existing social network infrastructure.
2. The algorithm has good scalability behavior, suitable for very large networks. Specifically, the algorithm presents manageable overhead in terms of message complexity, computation and storage, even for networks of millions of users. The algorithm can work asynchronously, as we do not assume synchronized clocks in a very large network.
3. The system maintains user privacy in the following sense: rating information is shared explicitly only among a user and her close social community. In particular, sharing does not involve a centralized third party.
4. The algorithm inherently favors highly-connected 'influential' users. This provides protection against spammers who wish to adversely influence ratings. With the rating system we propose, although spammers might occasionally get a foothold within a community, they cannot become influential.

Regarding the ranking of nodes in the network, we propose a novel way for distributed computation of matrix inverse based on the Gaussian Belief Propagation algorithm. This method is highly efficient for sparse graphs. Simulation results demonstrate convergence in about 5 rounds for a graph that includes about a million nodes. The message cost is minimal since each node sends one message containing two real numbers to its neighbor in each round.

We utilize our sparse matrix inversion method to compute a novel social influence ranking of nodes the social network we call "Spatial Ranking".

We show that our technique is general since it can compute other ranking methods like PageRank and Information Centrality with the same efficiency.

As a future work, we plan to implement a prototype of our algorithms to be used in real social networks like Nocturnal, demonstrating the feasibility of our approach in real settings.

Acknowledgements We would like to thank Yaakov Fernandess for his vision and support of the Nocturnal project, and Yair Weiss for being a great research mentor in the field of Gaussian Belief Propagation. We further like to thank Ciamak C. Moallemi and Benjamin Van-Roy for interesting discussions regarding the Consensus propagation algorithm.

References

[1] C. C. Moallemi and B. Van Roy, "Consensus Propagation," IEEE Transactions on Information Theory, Vol. 52, No. 11, pp. 4753-4766, 2006.

[2] Jason Johnson, Dmitry Malioutov, Alan Willsky, Walk-Sum Interpretation and Analysis of Gaussian Belief Propagation, In NIPS 05'.

[3] J.R. Shewchuk. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Technical Report CMU-CS-94-125, School of Computer Science, Carnegie Mellon University, 1994.

[4] Grinstead and Snell's Introduction to Probability. The CHANCE Project. Chapter 11 - Markov chains. `http://www.dartmouth.edu/ chance/teaching_aids/books_articles/probability_book/pdf.html`

[5] S. Brin and L. Page (1998). "The anatomy of a large-scale hypertextual Web search engine". Proceedings of the seventh international conference on World Wide Web 7, 107-117.

[6] Y. Weiss and W. T. Freeman. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. In NIPS-12, 1999

[7] Dmitry M. Malioutov, Jason K. Johnson, Alan S. Willsky. Walk-Sums and Belief Propagation in Gaussian Graphical Models. In JMLR 7(Oct):2031–2064, 2006.

[8] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The EigenTrust Algorithm for Reputation Management in P2P Networks. In Proceedings of the Twelfth International World Wide Web Conference, 2003.

[9] Jon M. Kleinberg, Authoritative sources in a hyperlinked environment, Journal of the ACM (JACM), v.46 n.5, p.604-632, Sept. 1999

[10] G. Elidan and I. McGraw and D. Koller, Residual Belief Propagation: Informed Scheduling for Asynchronous Message Passing, Proceedings of the Twenty-second Conference on Uncertainty in AI (UAI), Boston, Massachussetts, 2006

[11] K. Crammer and Y. Singer. Pranking with ranking. In Proceedings of the conference on Neural Information Processing Systems (NIPS), 2001.

[12] Amnon Shashua and Anat Levin. Ranking with large margin principle: Two approaches. In NIPS*14, 2003

[13] Jun Wang, Johan Pouwelse, Reginald Lagendijk and Marcel R. J. Reinders. Distributed Collaborative Filtering for Peer-to-Peer File Sharing Systems (2006), Proceedings of the 21st Annual ACM Symposium on Applied Computing(SAC06)

[14] Han Peng, Xie Bo, Yang Fan and Sheng Ruimin. A scalable P2P recommender system based on distributed collaborative filtering,Expert Systems with Applications, Vol. 27, No. 2. (August 2004), pp. 203-210.

[15] David Kempe, Jon Kleinberg, and Eva Tardos. Maximizing the spread of influence through a social network. In Proceedings of the Ninth International Conference on Knowledge Discovery and Data Mining, 2003.

[16] Yehuda Koren, Stephen C. North, Chris Volinsky: Measuring and extracting proximity in networks. In KDD 2006. p. 245-255

[17] Baruch Awerbuch, Boaz Patt-Shamir, David Peleg, and Mark Tuttle. Improved recommendation systems. In Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pages 1174-1183, January 2005.

[18] P. G. Doyle and J. L. Snell. Random Walks and Electrical Networks. The Mathematical Association of America, 1984. http://arxiv.org/abs/math.PR/0001057.

[19] Ulrik Brandes and Daniel Fleisch. Centrality Measures Based on Current Flow. STACS 2005, LNCS 3404, pp. 533544, 2005.

[20] Andras A. Benczr, Kroly Csalogny and Tams Sarls. On the feasibility of low-rank approximation for personalized PageRank. In: The 14th Int'l World Wide Web Conference (WWW2005, poster), 2005.

[21] `http://research.microsoft.com/research/sv/Nocturnal/default.aspx`